Minimum Average Distance Triangulations

László Kozma

Saarland University, Saarbrücken, Germany

European Symposium on Algorithms
10th September 2012

Weighted, undirected graph $G = (V, E, w)$.

$d_G(u, v)$: length of the shortest $u - v$ path.

Average distance:

$$\mathcal{A}(G) = \frac{1}{\binom{|V|}{2}} \sum_{\{x,y\} \subseteq V} d_G(u, v).$$

Weighted, undirected graph $G = (V, E, w)$.

$d_G(u, v)$: length of the shortest $u - v$ path.

Average distance:

$$\mathcal{A}(G) = \frac{1}{\binom{|V|}{2}} \sum_{\{x,y\} \subseteq V} d_G(u, v).$$

Total distance (H. Wiener, 1947):

$$\mathcal{W}(G) = \sum_{\{x,y\} \subseteq V} d_G(u, v).$$

Introduction

Network design (D.S. Johnson et al., 1978):

- Given $G = (V, E, w)$, find a spanning subgraph T,

  such as to minimize $$\mathcal{W}(T) = \sum_{\{x,y\} \subseteq V} d_T(x, y).$$

Introduction

Network design (D.S. Johnson et al., 1978):

- Given $G = (V, E, w)$, find a spanning subgraph T,

  such as to minimize $\quad \mathcal{W}(T) = \sum_{\{x,y\} \subseteq V} d_T(x, y)$.

- Budget constraint: $\sum_{e \in T} w(e) \leq B$.

Network design (D.S. Johnson et al., 1978):

- Given $G = (V, E, w)$, find a spanning subgraph T,

  such as to minimize $\quad \mathcal{W}(T) = \sum_{\{x,y\} \subseteq V} d_T(x, y)$.

- Budget constraint: $\sum_{e \in T} w(e) \leq B$.

- NP-hard, even with $w = 1$, and $B = |V| - 1$ (spanning tree).

A similar-looking *geometric* problem:

- Given $G = (V, E, w)$ embedded in the plane, A similar-looking *geometric* problem:

- Given $G = (V, E, w)$ embedded in the plane,

  find a spanning subgraph T,

  such as to minimize $\quad \mathcal{W}(T) = \sum_{\{x,y\} \subseteq V} d_T(x, y)$.

Introduction

A similar-looking *geometric* problem:

- Given $G = (V, E, w)$ embedded in the plane,

  find a spanning subgraph T,

  such as to minimize $\mathcal{W}(T) = \sum_{\{x,y\} \subseteq V} d_T(x, y)$.

- Constraint: T is non-crossing.

Introduction

A similar-looking *geometric* problem:

- Given $G = (V, E, w)$ embedded in the plane,

  find a spanning subgraph T,

  such as to minimize $\mathcal{W}(T) = \sum_{\{x,y\} \subseteq V} d_T(x, y)$.

- Constraint: T is non-crossing.
- No budget on total weight $\implies$ T is maximal non-crossing.

Minimum Average Distance Triangulation

Two variants:

Minimum Average Distance Triangulation

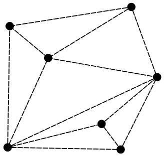
Two variants:

1. $S = \{p_1, p_2, \ldots, p_n\}$ points in $\mathbb{R}^2$,
   G is complete graph on S,
   $w : \binom{S}{2} \to \mathbb{R}$ pairwise distances (weights).
   $\implies T$ is a triangulation of S.

Minimum Average Distance Triangulation

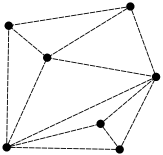
Two variants:

1. $S = \{p_1, p_2, \ldots, p_n\}$ points in $\mathbb{R}^2$,
   G is complete graph on S,
   $w : \binom{S}{2} \rightarrow \mathbb{R}$ pairwise distances (weights).
   $\implies$ T is a triangulation of S.



Minimum Average Distance Triangulation

Two variants:

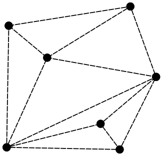


1. $S = \{p_1, p_2, \ldots, p_n\}$ points in $\mathbb{R}^2$,
   G is complete graph on S,
   $w : \binom{S}{2} \to \mathbb{R}$ pairwise distances (weights).
   $\implies$ T is a triangulation of S.
2. $P = (p_1, \ldots, p_n)$ a simple polygon,
   G consists of boundary and diagonal edges of P,
   $w : \binom{P}{2} \to \mathbb{R}$ pairwise distances (weights).
   $\implies$ T is a triangulation of the interior of P.

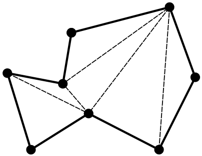
Minimum Average Distance Triangulation

Two variants:

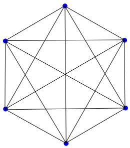
1. $S = \{p_1, p_2, \ldots, p_n\}$ points in $\mathbb{R}^2$,
   G is complete graph on S,
   $w : \binom{S}{2} \to \mathbb{R}$ pairwise distances (weights).
   $\implies$ T is a triangulation of S.



2. $P = (p_1, \ldots, p_n)$ a simple polygon,
   G consists of boundary and diagonal edges of P,
   $w : \binom{P}{2} \to \mathbb{R}$ pairwise distances (weights).
   $\implies$ T is a triangulation of the interior of P.

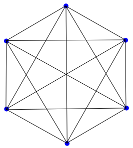


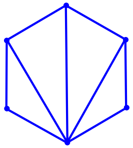
Example:
Regular hexagon, Euclidean weights.



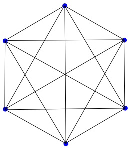
Example:

Regular hexagon, Euclidean weights.

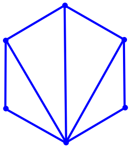




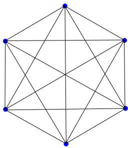
Example:
Regular hexagon, Euclidean weights.



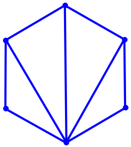
Minimum Average Distance Triangulation:
$\mathcal{W} \approx 24.93$, $\sum w(e) \approx 11.46$



Example:
Regular hexagon, Euclidean weights.

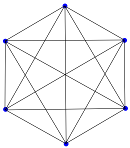


Minimum Average Distance Triangulation:
$\mathcal{W} \approx 24.93$, $\sum w(e) \approx 11.46$

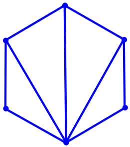


Minimum Weight Triangulation:
$\mathcal{W} \approx 25.39$, $\sum w(e) \approx 11.20$

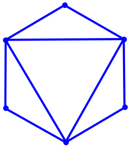
Example:
Regular hexagon, Euclidean weights.



Minimum Average Distance Triangulation:
$\mathcal{W} \approx 24.93$, $\sum w(e) \approx 11.46$



Minimum Weight Triangulation:
$\mathcal{W} \approx 25.39$, $\sum w(e) \approx 11.20$



Summary of results

$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\} \left.\begin{array}{r}\text{unit weights}\\ \text{Euclidean weights}\\ \text{arbitrary weights}\end{array}\right\} \quad \min_T \sum_{x,y} d_T(x,y).$$

Summary of results

$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\}\ \left.\begin{array}{r}\text{unit weights}\\ \text{Euclidean weights}\\ \text{arbitrary weights}\end{array}\right\}\quad \min_T \sum_{x,y} d_T(x,y).$$

1. Point sets, arbitrary (semimetric) weights: NP-complete

Summary of results

$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\} \left.\begin{array}{r}\text{unit weights}\\ \text{Euclidean weights}\\ \text{arbitrary weights}\end{array}\right\} \quad \min_T \sum_{x,y} d_T(x,y).$$

1. Point sets, arbitrary (semimetric) weights: NP-complete
2. Point sets, convex polygons, unit weights: trivial

Summary of results

$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\}\ \left.\begin{array}{r}\text{unit weights}\\ \text{Euclidean weights}\\ \text{arbitrary weights}\end{array}\right\}\quad \min_T \sum_{x,y} d_T(x,y).$$

1. Point sets, arbitrary (semimetric) weights: NP-complete
2. Point sets, convex polygons, unit weights: trivial
3. Arbitrary simple polygons, unit weights: $\mathcal{O}(n^{11})$

Summary of results

$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\}\ \begin{array}{r}\text{unit weights}\\ \text{Euclidean weights}\\ \text{arbitrary weights}\end{array}\left.\vphantom{\begin{array}{r}a\\a\\a\end{array}}\right\}\quad \min_T \sum_{x,y} d_T(x,y).$$

1. Point sets, arbitrary (semimetric) weights: NP-complete
2. Point sets, convex polygons, unit weights: trivial
3. Arbitrary simple polygons, unit weights: $\mathcal{O}(n^{11})$

Metric (e.g. Euclidean) weights: Open

1. Hardness result

MADT: point set $S \subseteq \mathbb{R}^2$, weights $w : S^2 \to \mathbb{R}$.

$$w \ \ \text{semimetric}: \ \forall x, y \in S : \begin{cases} w(x,y) \geq 0 \\ w(x,y) = 0 \ \text{ iff } \ x = y \\ w(x,y) = w(y,x) \end{cases}$$

1. Hardness result

MADT: point set $S \subseteq \mathbb{R}^2$, weights $w : S^2 \to \mathbb{R}$.

$$w \ \text{semimetric} : \ \forall x, y \in S : \begin{cases} w(x,y) \geq 0 \\ w(x,y) = 0 \ \text{ iff } \ x = y \\ w(x,y) = w(y,x) \end{cases}$$

Decision problem: given a threshold $\mathcal{W}^\star \in \mathbb{R}$, is there a triangulation T of S, such that:

$$\mathcal{W}(T) = \sum_{x,y \in S} d_T(x,y) \leq \mathcal{W}^\star$$

MADT $\in$ **NP**: use an all-pairs-shortest-path algorithm.

1. Hardness result

MADT $\in$ **NP**: use an all-pairs-shortest-path algorithm.

NP-hardness: gadget-based reduction from **Planar3SAT** [Lichtenstein, 1982]

1. Hardness result

MADT $\in$ **NP**: use an all-pairs-shortest-path algorithm.

NP-hardness: gadget-based reduction from **Planar3SAT** [Lichtenstein, 1982]

PLANAR 3SAT $\longrightarrow$ MAD TRIANGULATION

PLANAR 3SAT	MAD TRIANGULATION
• 3-CNF formula (variables, clauses) • planar embedding	• point set S • weights w • threshold $\mathcal{W}^\star$

$\exists$ satisfying assignment $\longleftrightarrow$ $\exists T$ s.t. $\mathcal{W}(T) \leq \mathcal{W}^\star$

2. Unit weights: the easy case

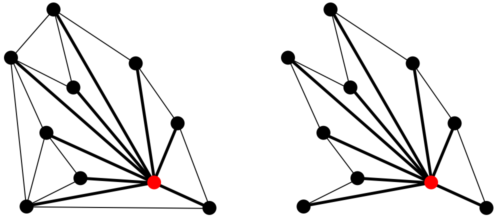
$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\}\ \text{one-vertex-visible}$$

$\mathcal{W}$ is minimized by the fan triangulation:

2. Unit weights: the easy case

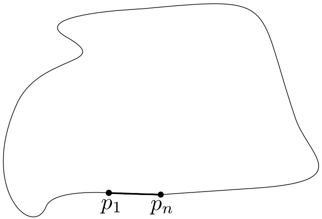
$$\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\}\ \text{one-vertex-visible}$$

$\mathcal{W}$ is minimized by the fan triangulation:



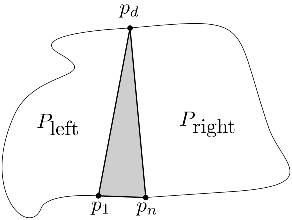
3. Unit weights: arbitrary polygons

Polygon $P = (p_1, \ldots, p_n)$ that does not admit a fan triangulation:



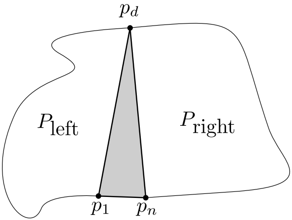
3. Unit weights: arbitrary polygons

Polygon $P = (p_1, \ldots, p_n)$ that does not admit a fan triangulation:



3. Unit weights: arbitrary polygons

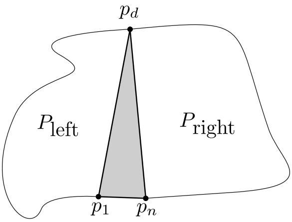
Polygon $P = (p_1, \ldots, p_n)$ that does not admit a fan triangulation:



General approach: *guess* a triangle that is part of the solution, decompose and recursively solve the problem ($\to$ dynamic programming).

3. Unit weights: arbitrary polygons

Polygon $P = (p_1, \ldots, p_n)$ that does not admit a fan triangulation:



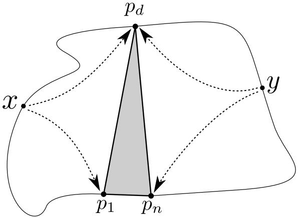
General approach: *guess* a triangle that is part of the solution, decompose and recursively solve the problem ($\rightarrow$ dynamic programming).

Problem: cost is not easy to decompose. How to deal with cross-distances?

Question: How to deal with cross-distances?

Let $x \in P_{left}$ and $y \in P_{right}$.

How does the path $x \leftrightarrow y$ cross the triangle $(p_1 p_d p_n)$?

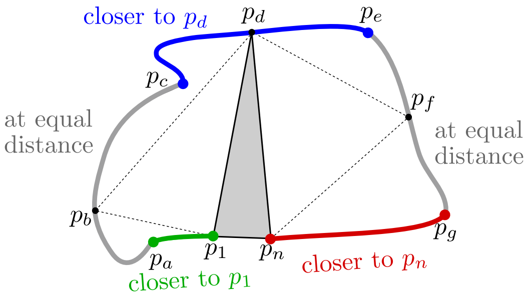


It depends on where x and y are.

Question: How to deal with cross-distances?

We assume the polygon is triangulated.

Observation: Vertices can be grouped into contiguous blocks.

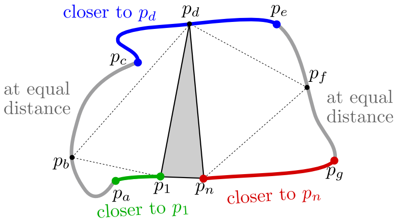


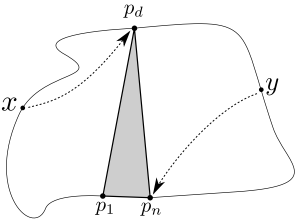
$x \in P_L$ (left part), $y \in P_R$ (right part)

Depending on where x and y fall, we can tell how $x \leftrightarrow y$ crosses the triangle, so we can decompose $d_T(x, y)$:

$x \in P_L$ (left part), $y \in P_R$ (right part)

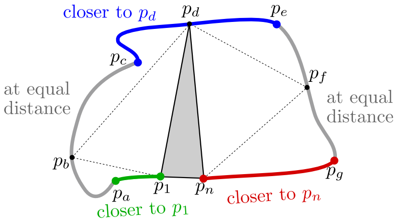
Depending on where x and y fall, we can tell how $x \leftrightarrow y$ crosses the triangle, so we can decompose $d_T(x, y)$:

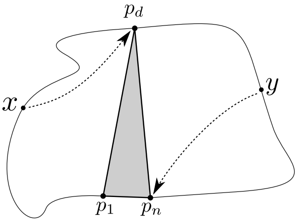




$x \in P_L$ (left part), $y \in P_R$ (right part)

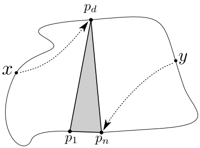
Depending on where x and y fall, we can tell how $x \leftrightarrow y$ crosses the triangle, so we can decompose $d_T(x, y)$:



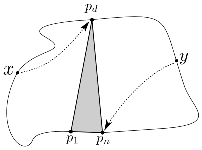


Let $\phi = d_T(x, p_d) + d_T(y, p_n)$. Then:

$$d_T(x, y) = \begin{cases} \phi - 1 & \text{if } y \in [p_d, p_e] \\ \phi + 1 & \text{if } y \in [p_g, p_n] \text{ and } x \in (p_a, p_d] \\ \phi & \text{otherwise .} \end{cases}$$

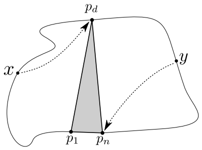


We need an extra term to accumulate distances from endpoints.
Solve a more general problem:



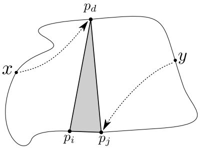
We need an extra term to accumulate distances from endpoints.
Solve a more general problem:

$$\mathcal{W}_{\mathrm{EXT}}(T, \alpha) = \sum_{x,y \in S} d_T(x, y)$$

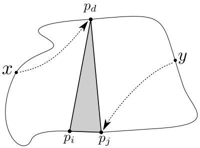


We need an extra term to accumulate distances from endpoints.
Solve a more general problem:

$$\mathcal{W}_{\mathrm{EXT}}(T, \alpha) = \sum_{x,y \in S} d_T(x, y) + \alpha \sum_{x \in S} d_T(x, p_n).$$

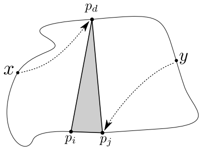


We need an extra term to accumulate distances from endpoints.
Solve a more general problem:



We need an extra term to accumulate distances from endpoints.
Solve a more general problem:

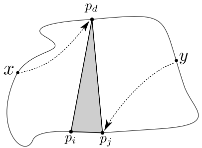
$$\mathcal{W}_{\mathrm{EXT}}(T,\alpha)\Big|_i^j = \sum_{x,y\in[p_i,p_j]} d_T(x,y) + \alpha \sum_{x\in[p_i,p_j]} d_T(x,p_j).$$



We need an extra term to accumulate distances from endpoints.
Solve a more general problem:

$$\mathcal{W}_{\mathrm{EXT}}(T,\alpha)\Big|_i^j = \sum_{x,y\in[p_i,p_j]} d_T(x,y) + \alpha \sum_{x\in[p_i,p_j]} d_T(x,p_j).$$

Minimizing $\mathcal{W}_{\mathrm{EXT}}(T,0)\Big|_1^n$ solves the initial problem.



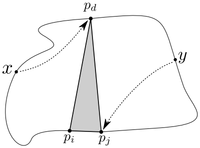
We need an extra term to accumulate distances from endpoints. Solve a more general problem:

$$\mathcal{W}_{\mathrm{EXT}}(T,\alpha)\Big|_i^j = \sum_{x,y\in[p_i,p_j]} d_T(x,y) + \alpha \sum_{x\in[p_i,p_j]} d_T(x,p_j).$$

Minimizing $\mathcal{W}_{\mathrm{EXT}}(T,0)\Big|_1^n$ solves the initial problem.

Using the formula for $d_T(x,y)$ we can express $\mathcal{W}_{\mathrm{EXT}}$ recursively:

$$\mathcal{W}_{\mathrm{EXT}}(T,\alpha)\Big|_i^j = \mathcal{W}_{\mathrm{EXT}}(T,\ \dots\ )\Big|_i^d + \mathcal{W}_{\mathrm{EXT}}(T,\ \dots\ )\Big|_d^j$$
$$+\ \dots$$



We need an extra term to accumulate distances from endpoints.
Solve a more general problem:

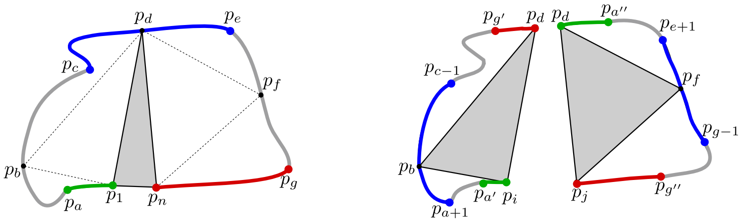
$$\mathcal{W}_{\mathrm{EXT}}(T,\alpha)\Big|_i^j = \sum_{\substack{x,y\in[p_i,p_j]\\ x<y}} d_T(x,y) + \alpha \sum_{x\in[p_i,p_j]} d_T(x,p_j).$$

Minimizing $\mathcal{W}_{\mathrm{EXT}}(T,0)\Big|_1^n$ solves the initial problem.

Using the formula for $d_T(x,y)$ we can express $\mathcal{W}_{\mathrm{EXT}}$ recursively:

$$\mathcal{W}_{\mathrm{EXT}}(T,\alpha)\Big|_i^j = \mathcal{W}_{\mathrm{EXT}}(T,\alpha+j-d)\Big|_i^d + \mathcal{W}_{\mathrm{EXT}}(T,\alpha+d-i)\Big|_d^j$$
$$+ (\alpha+j-g+1)(d-a-1) + (e-d+1)(i-d).$$

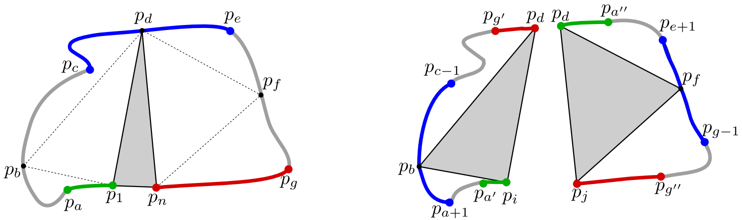
We need to ensure that the constraints on the indices are respected:



Observation: Assume T contains the triangles $p_i p_d p_j$ and $p_i p_b p_d$.

(a) a is the largest index s.t. $d_T(p_a, p_i) < d_T(p_a, p_d)$ iff $a + 1$ is the smallest index s.t. $d_T(p_{a+1}, p_b) < d_T(p_{a+1}, p_i)$.

We need to ensure that the constraints on the indices are respected:

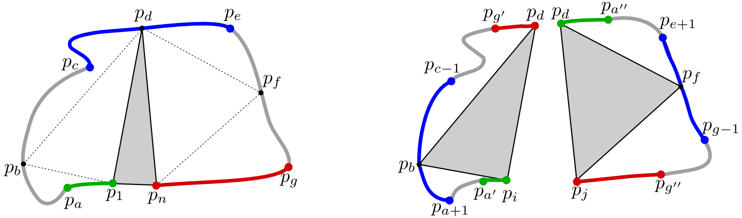


Observation: Assume T contains the triangles $p_i p_d p_j$ and $p_i p_b p_d$.

(a) a is the largest index s.t. $d_T(p_a, p_i) < d_T(p_a, p_d)$ iff $a+1$ is the smallest index s.t. $d_T(p_{a+1}, p_b) < d_T(p_{a+1}, p_i)$.

(b) c is the smallest index s.t. $d_T(p_c, p_d) < d_T(p_c, p_i)$ iff $c-1$ is the largest index s.t. $d_T(p_{c-1}, p_b) < d_T(p_{c-1}, p_d)$.

We need to ensure that the constraints on the indices are respected:



Observation: Assume T contains the triangles $p_i p_d p_j$ and $p_i p_b p_d$.

(a) a is the largest index s.t. $d_T(p_a, p_i) < d_T(p_a, p_d)$ iff $a + 1$ is the smallest index s.t. $d_T(p_{a+1}, p_b) < d_T(p_{a+1}, p_i)$.

(b) c is the smallest index s.t. $d_T(p_c, p_d) < d_T(p_c, p_i)$ iff $c - 1$ is the largest index s.t. $d_T(p_{c-1}, p_b) < d_T(p_{c-1}, p_d)$.

(c) analogous relations on the right side

Putting it all together:

procedure $EXT\left((p_i, \ldots, p_j),\ p_a, p_c, p_e, p_g,\ \alpha\right)$:

if $(a = i)$ **and** $(c = e = i + 1)$ **and** $(g = j = i + 2)$:

return $(3 + 2\alpha)$; /* *the polygon has only three vertices* */

else:

$$\textbf{return} \min_{\substack{p_d, p'_a, p'_g, p''_a, p''_g: \\ i \le a' \le a+1 \le c-1 \le g' \le d \\ d \le a'' \le e+1 \le g-1 \le g'' \le j \\ p_i \leftrightarrow p_d \leftrightarrow p_j}} \Big\{ EXT\left((p_i, \ldots, p_d),\ p'_a, p_{a+1}, p_{c-1}, p'_g,\ \alpha + j - d\right) + EXT\left((p_d, \ldots, p_j),\ p''_a, p_{e+1}, p_{g-1}, p''_g,\ \alpha + d - i\right) + (\alpha + j - g + 1)(d - a - 1) + (e - d + 1)(i - d) \Big\};$$

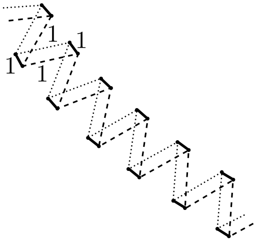
Summary of results

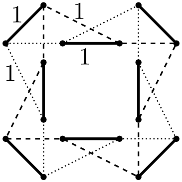
$$\left.\left.\begin{array}{r}\text{point sets}\\ \text{polygons}\end{array}\right\} \begin{array}{r}\text{unit weights}\\ \text{Euclidean weights}\\ \text{arbitrary weights}\end{array}\right\} \quad \min_T \sum_{x,y} d_T(x,y).$$

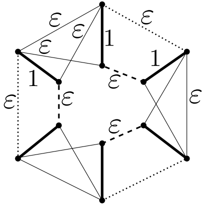
1. Point sets, arbitrary (semimetric) weights: NP-complete
2. Point sets, convex polygons, unit weights: trivial
3. Arbitrary simple polygons, unit weights: $\mathcal{O}(n^{11})$

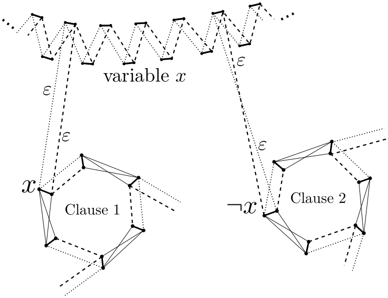
Metric (e.g. Euclidean) weights: Open

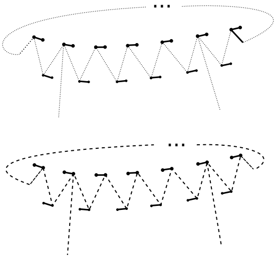
Thank you for your attention.

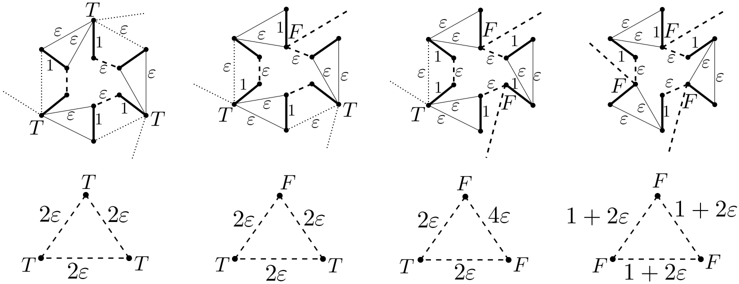

1
1
1
1


1
1
1
1


ε
ε
ε
ε
1
1
1
ε
ε
ε
ε
ε


variable x
ε
ε
ε
ε
x
$\neg x$
Clause 1
Clause 2




T
2ε
2ε
T
2ε
T
F
2ε
2ε
T
2ε
T
F
2ε
4ε
T
2ε
F
F
1 + 2ε
1 + 2ε
F
1 + 2ε
F